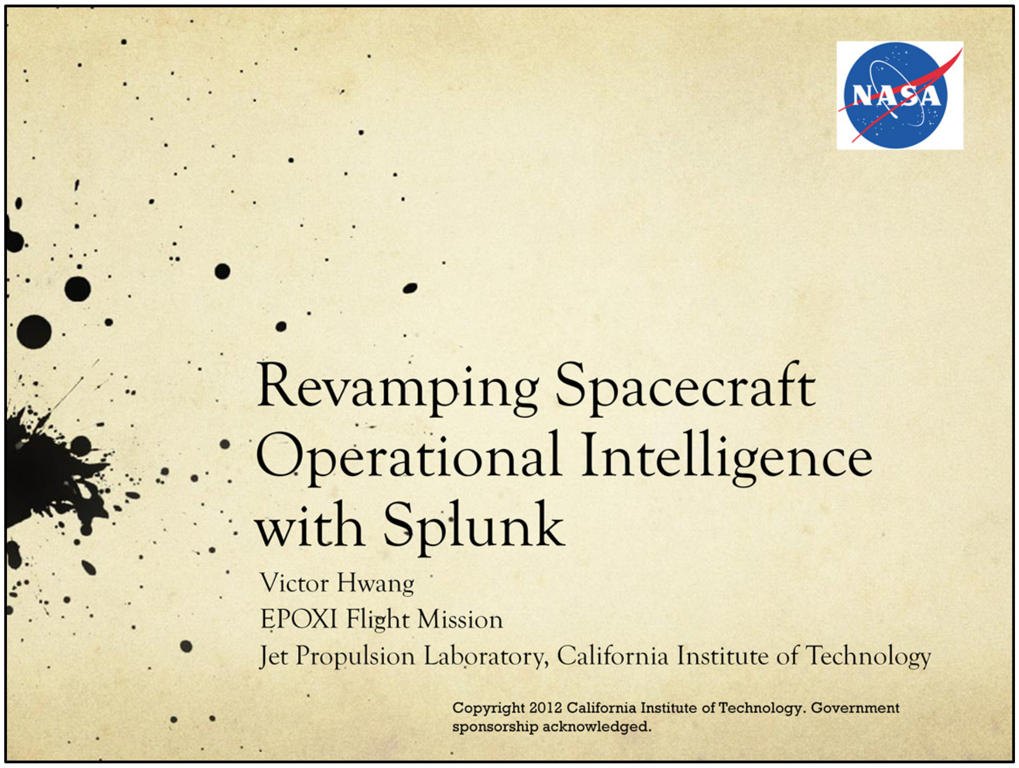
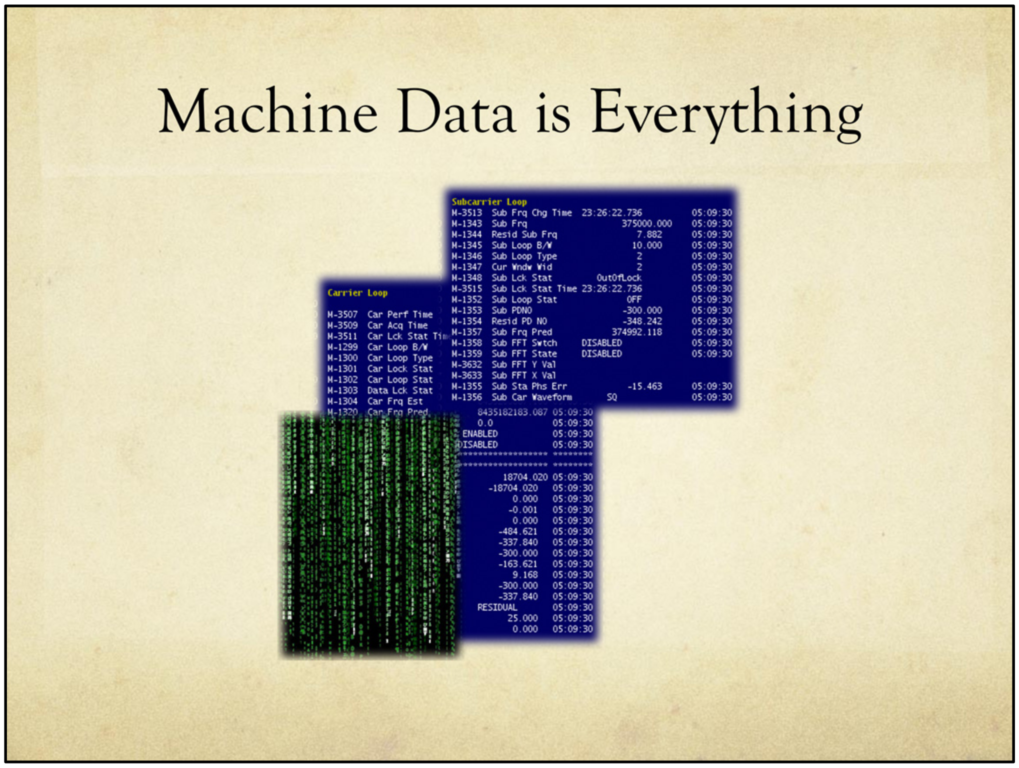


Revamping Spacecraft Operational Intelligence with Splunk

Victor Hwang
EPOXI Flight Mission
Jet Propulsion Laboratory, California Institute of Technology

Hi everybody. Today I'm going to talk about EPOXI's experience with a Big Data tool, Splunk. The goal of this experiment was to try and improve our team's operational intelligence, or in another words, the ability for us to see trends and high level information in the machine data

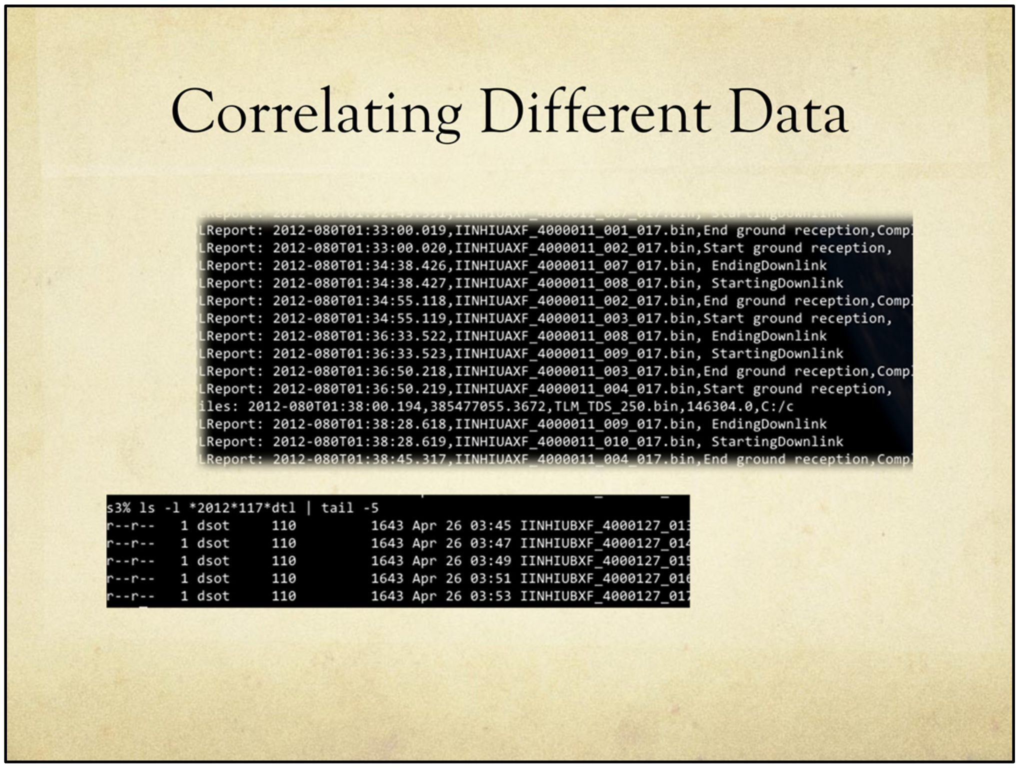


Machine Data is Everything

```
Subcarrier Loop
M-3513  Sub Frq Chg Time  23:26:22.736              05:09:30
M-1343  Sub Frq                       375000.000    05:09:30
M-1344  Resid Sub Frq                      7.882    05:09:30
M-1345  Sub Loop B/W                      10.000    05:09:30
M-1346  Sub Loop Type                 2             05:09:30
M-1347  Cur Wndw Wid                  2             05:09:30
M-1348  Sub Lck Stat          OutOfLock             05:09:30
M-3515  Sub Lck Stat Time 23:26:22.736              05:09:30
M-1352  Sub Loop Stat               OFF             05:09:30
M-1353  Sub PDNO                        -300.000    05:09:30
M-1354  Resid PD NO                     -348.242    05:09:30
M-1357  Sub Frq Pred                  374992.118    05:09:30
M-1358  Sub FFT Swtch     DISABLED                  05:09:30
M-1359  Sub FFT State     DISABLED                  05:09:30
M-3632  Sub FFT Y Val
M-3633  Sub FFT X Val
M-1355  Sub Sta Phs Err                  -15.463    05:09:30
M-1356  Sub Car Waveform       SQ                   05:09:30
```

```
Carrier Loop

M-3507  Car Perf Time
M-3509  Car Acq Time
M-3511  Car Lck Stat Tim
M-1299  Car Loop B/W
M-1300  Car Loop Type
M-1301  Car Lock Stat
M-1302  Car Loop Stat
M-1303  Data Lck Stat
M-1304  Car Frq Est
M-1320  Car Frq Pred
        8435182183.087 05:09:30
        0.0            05:09:30
       ENABLED         05:09:30
       DISABLED        05:09:30
      **************** ********
      **************** ********
             18704.020 05:09:30
          -18704.020   05:09:30
               0.000   05:09:30
              -0.001   05:09:30
               0.000   05:09:30
            -484.621   05:09:30
            -337.840   05:09:30
            -300.000   05:09:30
            -163.621   05:09:30
               9.168   05:09:30
            -300.000   05:09:30
            -337.840   05:09:30
        RESIDUAL       05:09:30
              25.000   05:09:30
               0.000   05:09:30
```

- The reason I started down this path to begin with was because I'm quite new to mission operations. I joined the team about 2 years ago having worked mostly in web and software development, with an interest in data visualization.
- So when I began learning to fly the spacecraft, I was instantly overwhelmed by the sheer amount of data I had to filter through. In this image, you can see one of our telemetry displays. Unlike my very seasoned co-workers, I had a difficult time getting an overall understanding of what was going on just by glancing at these pages. If a problem occurred, I wouldn't notice until later because the important information would be hidden in pages and pages of data.
- . Once I became more familiar with the telemetry data and displays, I was still pretty inefficient at reading it quickly because I couldn't customize the interface to my liking, and the data was more low-level than I needed to see.



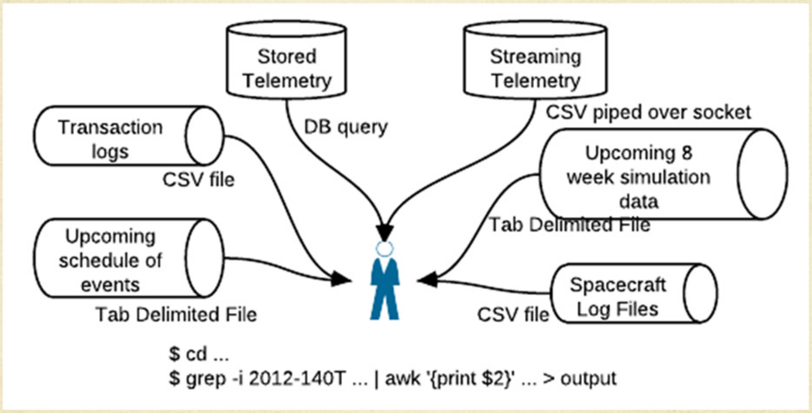
Correlating Different Data

```
LReport: 2012-080T01:33:00.019,IINHIUAXF_4000011_001_017.bin,End ground reception,Comp
LReport: 2012-080T01:33:00.020,IINHIUAXF_4000011_002_017.bin,Start ground reception,
LReport: 2012-080T01:34:38.426,IINHIUAXF_4000011_007_017.bin, EndingDownlink
LReport: 2012-080T01:34:38.427,IINHIUAXF_4000011_008_017.bin, StartingDownlink
LReport: 2012-080T01:34:55.118,IINHIUAXF_4000011_002_017.bin,End ground reception,Comp
LReport: 2012-080T01:34:55.119,IINHIUAXF_4000011_003_017.bin,Start ground reception,
LReport: 2012-080T01:36:33.522,IINHIUAXF_4000011_008_017.bin, EndingDownlink
LReport: 2012-080T01:36:33.523,IINHIUAXF_4000011_009_017.bin, StartingDownlink
LReport: 2012-080T01:36:50.218,IINHIUAXF_4000011_003_017.bin,End ground reception,Comp
LReport: 2012-080T01:36:50.219,IINHIUAXF_4000011_004_017.bin,Start ground reception,
iles: 2012-080T01:38:00.194,385477055.3672,TLM_TDS_250.bin,146304.0,C:/c
LReport: 2012-080T01:38:28.618,IINHIUAXF_4000011_009_017.bin, EndingDownlink
LReport: 2012-080T01:38:28.619,IINHIUAXF_4000011_010_017.bin, StartingDownlink
LReport: 2012-080T01:38:45.317,IINHIUAXF_4000011_004_017.bin,End ground reception,Comp
```

```
s3% ls -l *2012*117*dtl | tail -5
r--r--   1 dsot     110         1643 Apr 26 03:45 IINHIUBXF_4000127_01
r--r--   1 dsot     110         1643 Apr 26 03:47 IINHIUBXF_4000127_01
r--r--   1 dsot     110         1643 Apr 26 03:49 IINHIUBXF_4000127_01
r--r--   1 dsot     110         1643 Apr 26 03:51 IINHIUBXF_4000127_01
r--r--   1 dsot     110         1643 Apr 26 03:53 IINHIUBXF_4000127_01
```

I had trouble with other things beyond the telemetry system as well. when analyzing historic spacecraft data, I always wasted a lot of time collecting data. On top of that, I would spend a non-trivial amount of time scripting one-time-use tools to do a quick analysis. For example, there were times when I wanted to make sure our file downlink rate was what I expected. This is important in order to know how many files we can downlink in a given time. In order to do this, I would have to compare our prediction data with the timestamps of files. This picture shows the two formats of the information – a flat file with a bunch of text, and a long directory listing of when the files were actually received on the ground. Doing this analysis was a little annoying.

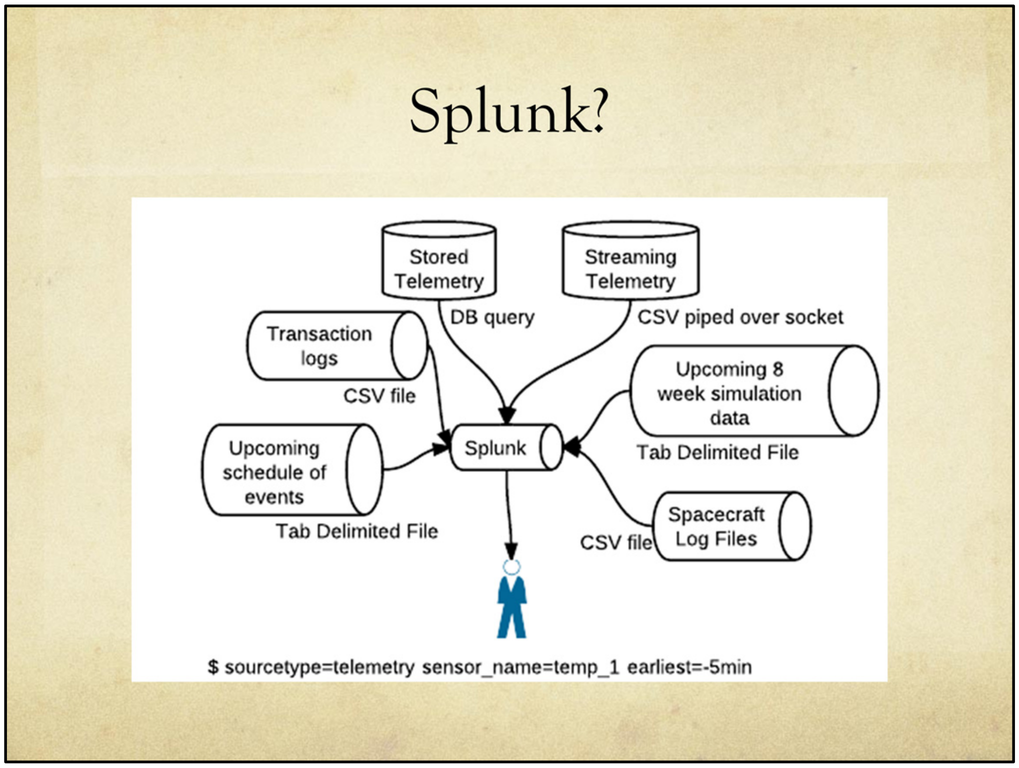
“Where do I find that data?”



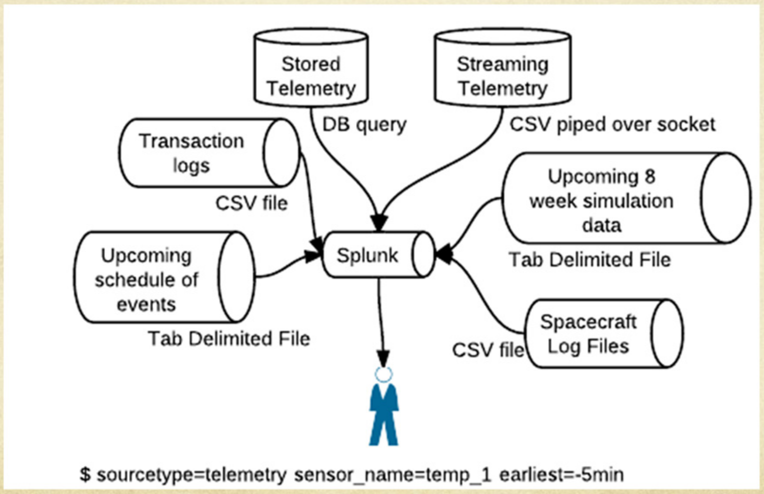
```
$ cd ...
$ grep -i 2012-140T ... | awk '{print $2}' ... > output
```

So, to put the problem in concrete terms, the trouble I had stemmed from the fact that our data was scattered all over the place in difficult to read formats. Beyond real time telemetry data, we have transaction logs, downlinked images, spacecraft logs, schedules, and more. These were stored on a number of different servers and databases, and all had varying formats.

And, as I said before, parsing this data always involved some serious command line munging.



Splunk?

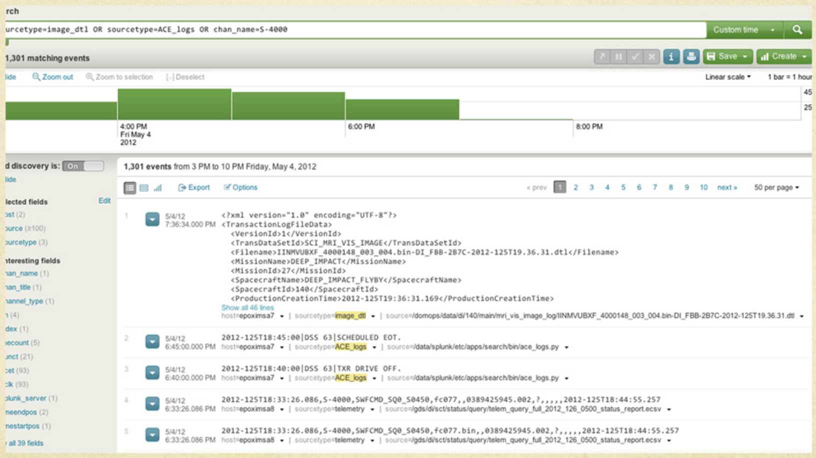


```
$ sourcetype=telemetry sensor_name=temp_1 earliest=-5min
```

So what is Splunk? Instead of giving the technical details, which you can find online, I'll tell you what it did for me. Splunk slapped everything into one place, with one uniform format, and gave me the ability to forget about all these annoying details of where it is, how to parse it, and all that. Instead, I only need to interact with Splunk to find the data I need. This sounds simple and obvious, but it's surprising what you can do once you all of your data is indexed in one place.

By having your data organized, querying becomes much easier. Let's say that I want to search telemetry for a sensor_name "temp_1" and to return all data that is at most five minutes old. And because Splunk can hook into a real-time stream, this data will always be up-to-date.

Data Mashup



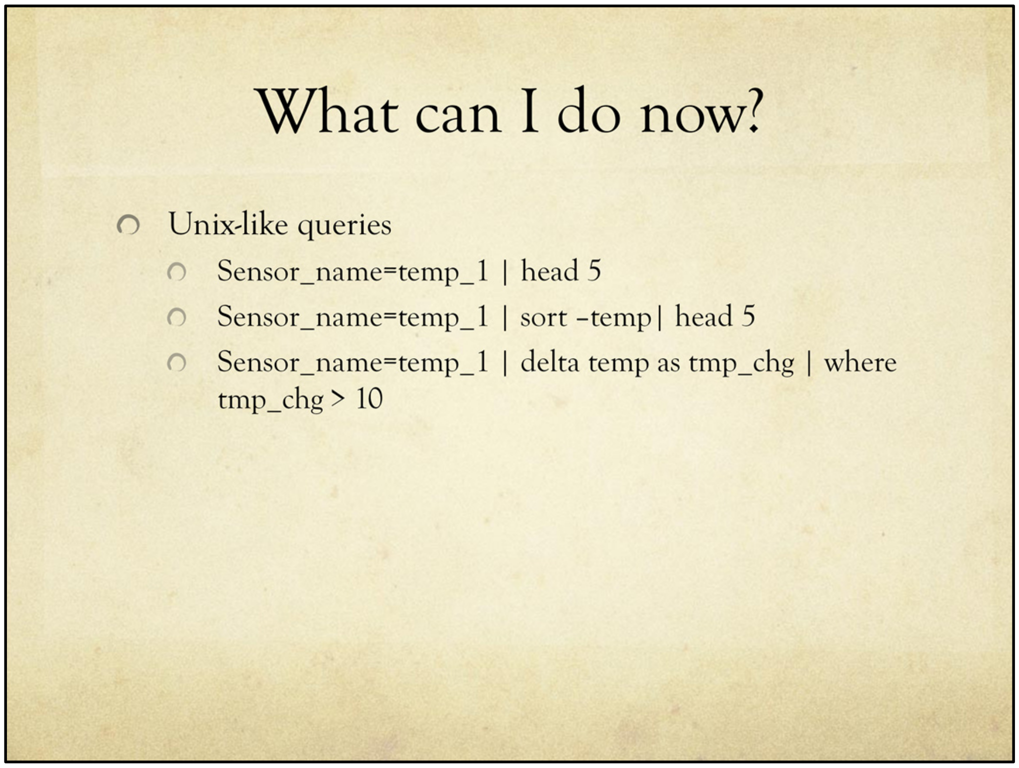
```
urcetype=image_dtl OR sourcetype=ACE_logs OR chan_name=S-4000
```

1,301 matching events

1,301 events from 3 PM to 10 PM Friday, May 4, 2012

```
1  5/4/12            <?xml version="1.0" encoding="UTF-8"?>
   7:36:34.000 PM    <TransactionLogFileData>
                       <VersionId>1</VersionId>
                       <TransDataSetId>SCI_MRI_VIS_IMAGE</TransDataSetId>
                       <Filename>IINMVUBXF_4000148_003_004.bin-DI_FBB-2B7C-2012-125T19.36.31.dtl</Filename>
                       <MissionName>DEEP_IMPACT</MissionName>
                       <MissionId>27</MissionId>
                       <SpacecraftName>DEEP_IMPACT_FLYBY</SpacecraftName>
                       <SpacecraftId>140</SpacecraftId>
                       <ProductionCreationTime>2012-125T19:36:31.169</ProductionCreationTime>
                     Show all 46 lines
                     host=epoximsa7 | sourcetype=image_dtl | source=/domops/data/di/140/main/mri_vis_image_log/IINMVUBXF_4000148_003_004.bin-DI_FBB-2B7C-2012-125T19.36.31.dtl

2  5/4/12            2012-125T18:45:00|DSS 63|SCHEDULED EOT.
   6:45:00.000 PM    host=epoximsa7 | sourcetype=ACE_logs | source=/data/splunk/etc/apps/search/bin/ace_logs.py

3  5/4/12            2012-125T18:40:00|DSS 63|TXR DRIVE OFF.
   6:40:00.000 PM    host=epoximsa7 | sourcetype=ACE_logs | source=/data/splunk/etc/apps/search/bin/ace_logs.py

4  5/4/12            2012-125T18:33:26.086,S-4000,SWFCMD_SQ0_S0450,fc077,,0389425945.002,?,,,,,2012-125T18:44:55.257
   6:33:26.086 PM    host=epoximsa8 | sourcetype=telemetry | source=/gds/di/sct/status/query/telem_query_full_2012_126_0500_status_report.ecsv

5  5/4/12            2012-125T18:33:26.086,S-4000,SWFCMD_SQ0_S0450,fc077.bin,,0389425945.002,?,,,,,2012-125T18:44:55.257
   6:33:26.086 PM    host=epoximsa8 | sourcetype=telemetry | source=/gds/di/sct/status/query/telem_query_full_2012_126_0500_status_report.ecsv
```

Extending the previous example, I can now aggregate all types of data into one view based in time. In this picture, I've got transaction logs, telemetry, and downlinked files all in one page, organized by time. Even though the raw data looks completely than this, I've defined interfaces that transform it into this uniform format. This gives me a more complete picture for the question "what was the spacecraft doing at this particular time?" And because querying data is simple, I can start with a big block of data and whiddle it down to what I need, rather than hunting around for the individual pieces of data that I need.



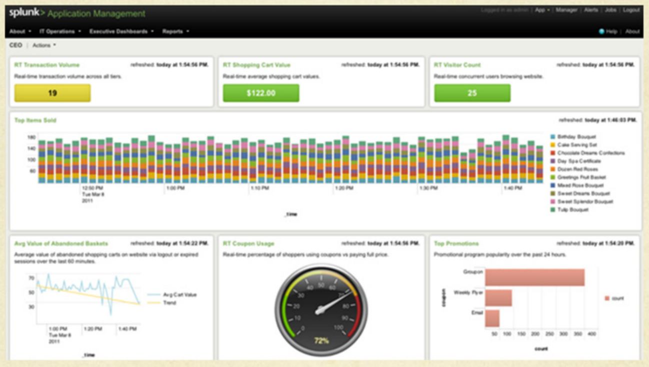
What can I do now?

- Unix-like queries
  - Sensor_name=temp_1 | head 5
  - Sensor_name=temp_1 | sort –temp| head 5
  - Sensor_name=temp_1 | delta temp as tmp_chg | where tmp_chg > 10

When we have all the data we need, we can begin widdling down the data with Splunk's Unix-like search syntax. These three examples highlights my trial-and-error attempts to find large temperature changes. I begin by showing the first 5 temperatures, only to find that they're sorted chronologically, rather than from highest temperatures to lowest temperatures. The next line shows sorting temperatures by their values, but I find that that's not really what I want either. I want to know the delta temperatures between readings. Looking through Splunk's user manual, I find the delta function, which lets me dynamically generate new information to use in my query. With that extra piece of information, I can now return only the telemetry readings where the temperature changed by at least 10.

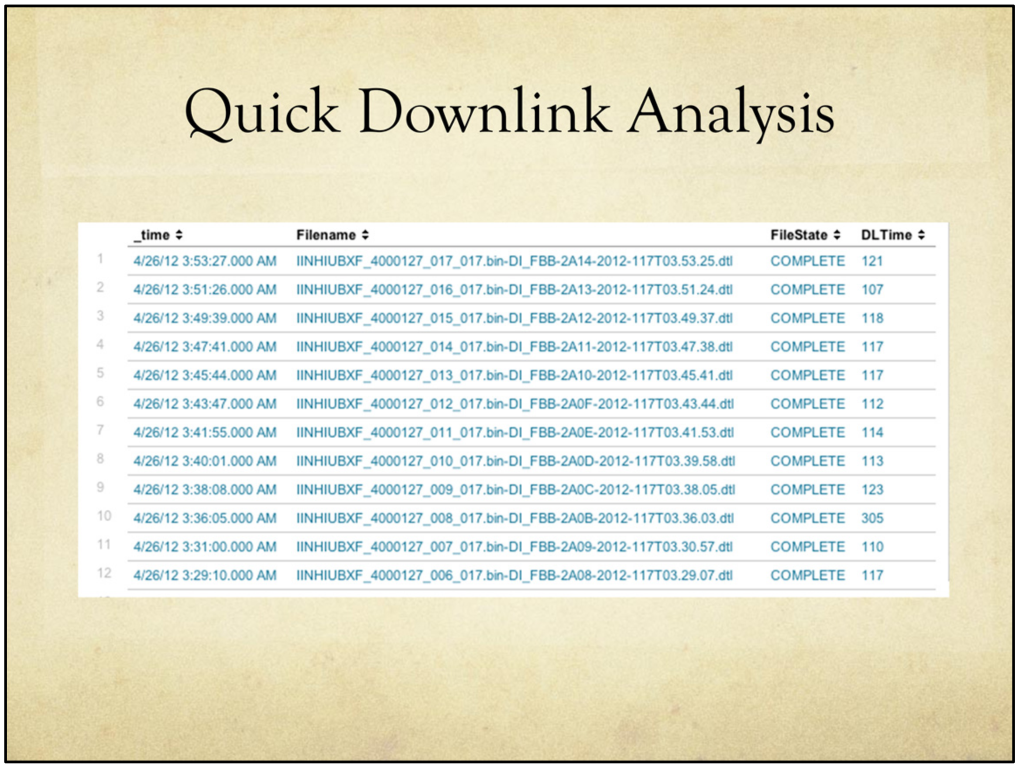
One other useful feature I'll mention is that all of these queries can be run through Splunk's API. So any scripting language you can think of can plug right in and make these queries. This gives us the ability to build a lot of new tools.

Dashboards/Telemetry Displays



With this query language, you can start building some pretty cool tools. If you look carefully, you'll notice this telemetry display, or dashboard, has absolutely nothing to do with spacecrafts. However, it's a good example of the user interface and visualizations Splunk can provide. This in-browser dashboard hooks right into Splunk's search system, and you can begin building some impressive displays with only a few clicks of a mouse.

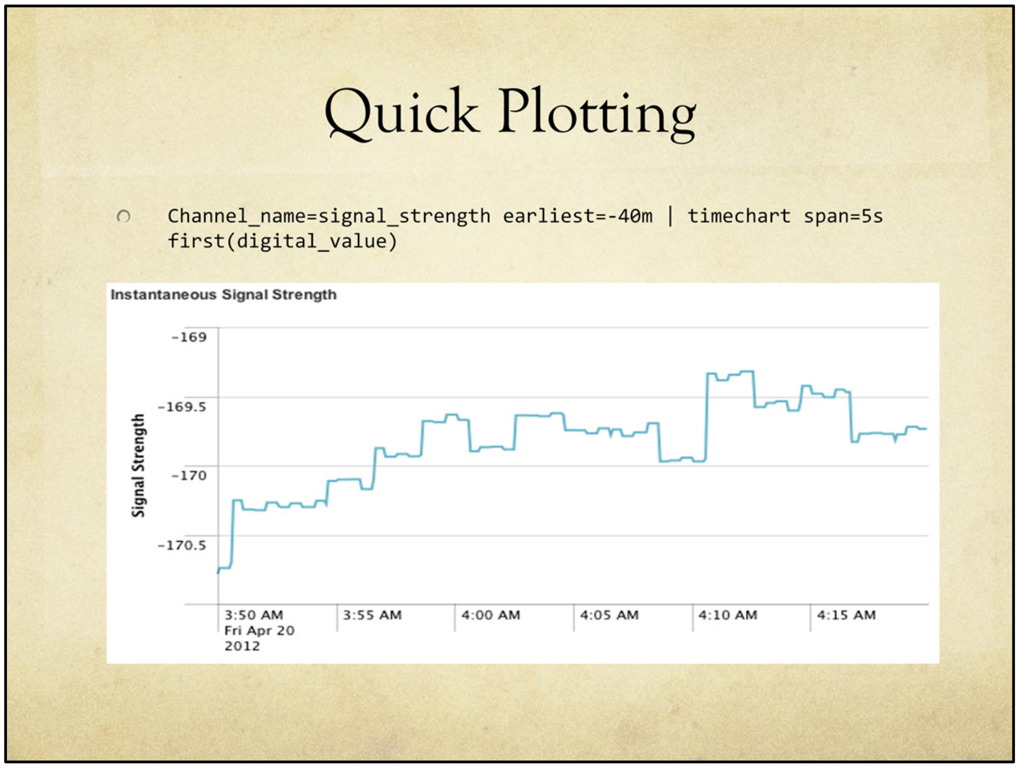
Now I'll show you some of my dashboards, which are not nearly as pretty as this.



Quick Downlink Analysis

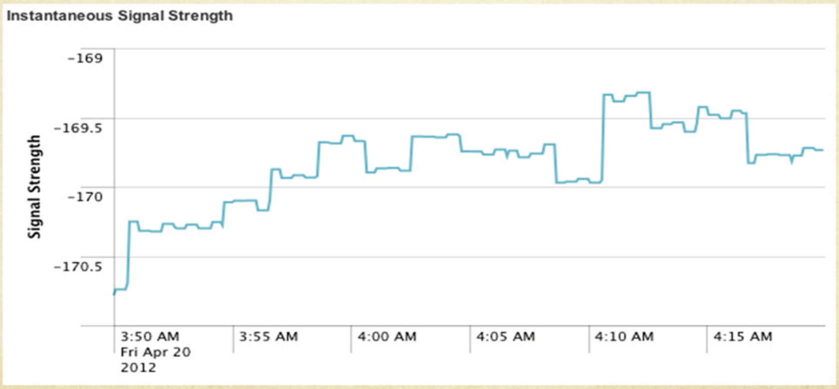
	_time	Filename	FileState	DLTime
1	4/26/12 3:53:27.000 AM	IINHIUBXF_4000127_017_017.bin-DI_FBB-2A14-2012-117T03.53.25.dtl	COMPLETE	121
2	4/26/12 3:51:26.000 AM	IINHIUBXF_4000127_016_017.bin-DI_FBB-2A13-2012-117T03.51.24.dtl	COMPLETE	107
3	4/26/12 3:49:39.000 AM	IINHIUBXF_4000127_015_017.bin-DI_FBB-2A12-2012-117T03.49.37.dtl	COMPLETE	118
4	4/26/12 3:47:41.000 AM	IINHIUBXF_4000127_014_017.bin-DI_FBB-2A11-2012-117T03.47.38.dtl	COMPLETE	117
5	4/26/12 3:45:44.000 AM	IINHIUBXF_4000127_013_017.bin-DI_FBB-2A10-2012-117T03.45.41.dtl	COMPLETE	117
6	4/26/12 3:43:47.000 AM	IINHIUBXF_4000127_012_017.bin-DI_FBB-2A0F-2012-117T03.43.44.dtl	COMPLETE	112
7	4/26/12 3:41:55.000 AM	IINHIUBXF_4000127_011_017.bin-DI_FBB-2A0E-2012-117T03.41.53.dtl	COMPLETE	114
8	4/26/12 3:40:01.000 AM	IINHIUBXF_4000127_010_017.bin-DI_FBB-2A0D-2012-117T03.39.58.dtl	COMPLETE	113
9	4/26/12 3:38:08.000 AM	IINHIUBXF_4000127_009_017.bin-DI_FBB-2A0C-2012-117T03.38.05.dtl	COMPLETE	123
10	4/26/12 3:36:05.000 AM	IINHIUBXF_4000127_008_017.bin-DI_FBB-2A0B-2012-117T03.36.03.dtl	COMPLETE	305
11	4/26/12 3:31:00.000 AM	IINHIUBXF_4000127_007_017.bin-DI_FBB-2A09-2012-117T03.30.57.dtl	COMPLETE	110
12	4/26/12 3:29:10.000 AM	IINHIUBXF_4000127_006_017.bin-DI_FBB-2A08-2012-117T03.29.07.dtl	COMPLETE	117

This image shows one dashboard element I made with only one search query. As I mentioned before, I often check to see how long it takes to downlink an image. Here, as the images come down in real-time, Splunk's system continuously refreshes a table containing each file and their downlink time. Again, this was formed with just a few search queries similar to those on the previous slide.

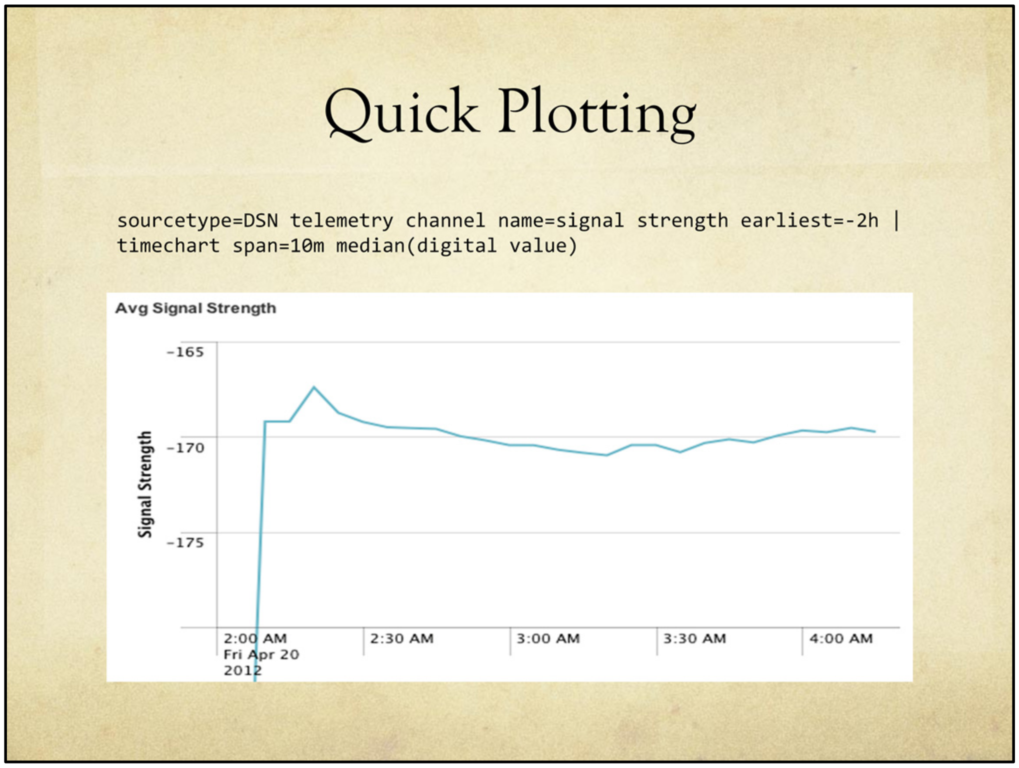


Quick Plotting

- `Channel_name=signal_strength earliest=-40m | timechart span=5s first(digital_value)`

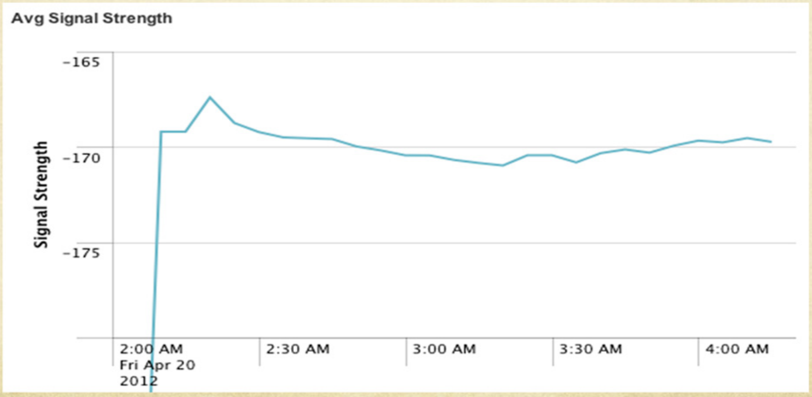


Splunk has a powerful graphing module that allows you to set up real-time plots very quickly. Here, we're plotting signal strength over time by specifying the period that we'd like to graph over (last 40 minutes) and taking one value every 5 seconds. The problem with this graph is that we cannot see the trend – the timespan is too short, and the data is too noisy. In our old system, this would take a bit of scripting to fix.



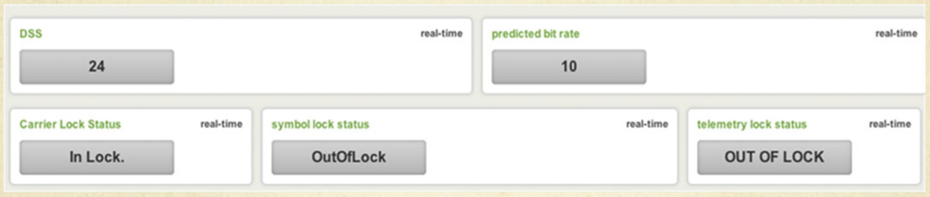
Quick Plotting

```
sourcetype=DSN telemetry channel name=signal strength earliest=-2h |
timechart span=10m median(digital value)
```



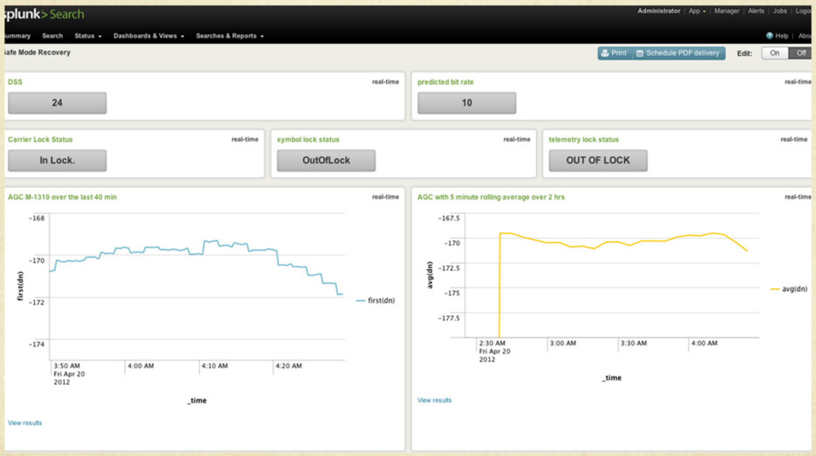
However, all it takes with Splunk is to alter the search syntax to span the past two hours, bucket the data into 10 minute chunks and take the median. Now we see the trend much more clearly.

Building Telemetry Displays



Along with the graphing modules, Splunk also has intuitive user interfaces for building "dashboards", which are equivalent to telemetry displays. While this telemetry display isn't novel by any means, it was built in less than five minutes with a simple point-and-click interface. Because Splunk also has LDAP authentication built in, each user can build their own unique telemetry display or pick from a global set of displays. Because these are all in-browser displays, they can be pulled up virtually anywhere with any system.

Putting it together...


splunk> Search
Administrator | App | Manager | Alerts | Jobs | Logout
Summary Search Status Dashboards & Views Searches & Reports
Help | About
Print
Schedule PDF delivery
Edit: On Off
AGC M-1310 over the last 40 min
first(dn)
3:50 AM Fri Apr 20 2012
4:00 AM
4:10 AM
4:20 AM
_time
AGC with 5 minute rolling average over 2 hrs
avg(dn)
2:30 AM Fri Apr 20 2012
3:00 AM
3:30 AM
4:00 AM
View results

Safe Mode Recovery

DSS (real-time): 24

predicted bit rate (real-time): 10

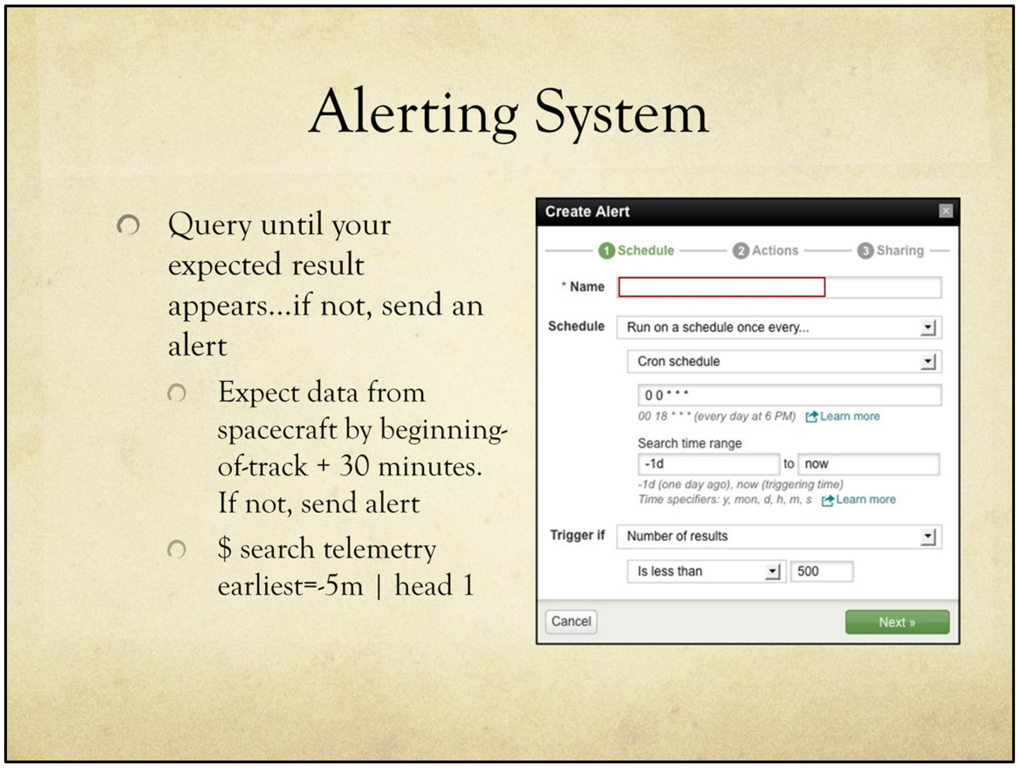
Carrier Lock Status (real-time): In Lock.

symbol lock status (real-time): OutOfLock

telemetry lock status (real-time): OUT OF LOCK

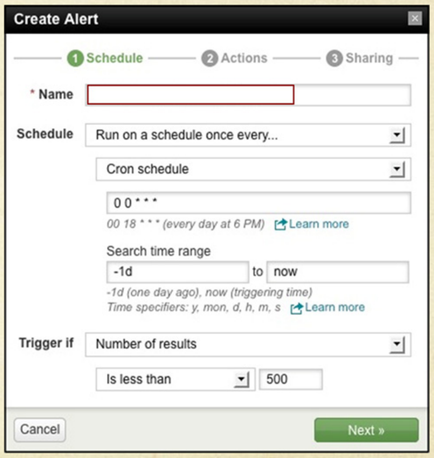
So here is a complete picture of one of my dashboards, with the elements that I described earlier. Not nearly as pretty as the promotional screenshot, but considering this took me 10 minutes to make, I'm pretty pleased.

As a side note, one useful feature that our current system lacks is back-fill. Meaning, when I pull up this page, it will populate graphs immediately without having to wait for more data to stream it.



Alerting System

- Query until your expected result appears…if not, send an alert
  - Expect data from spacecraft by beginning-of-track + 30 minutes. If not, send alert
  - $ search telemetry earliest=-5m | head 1

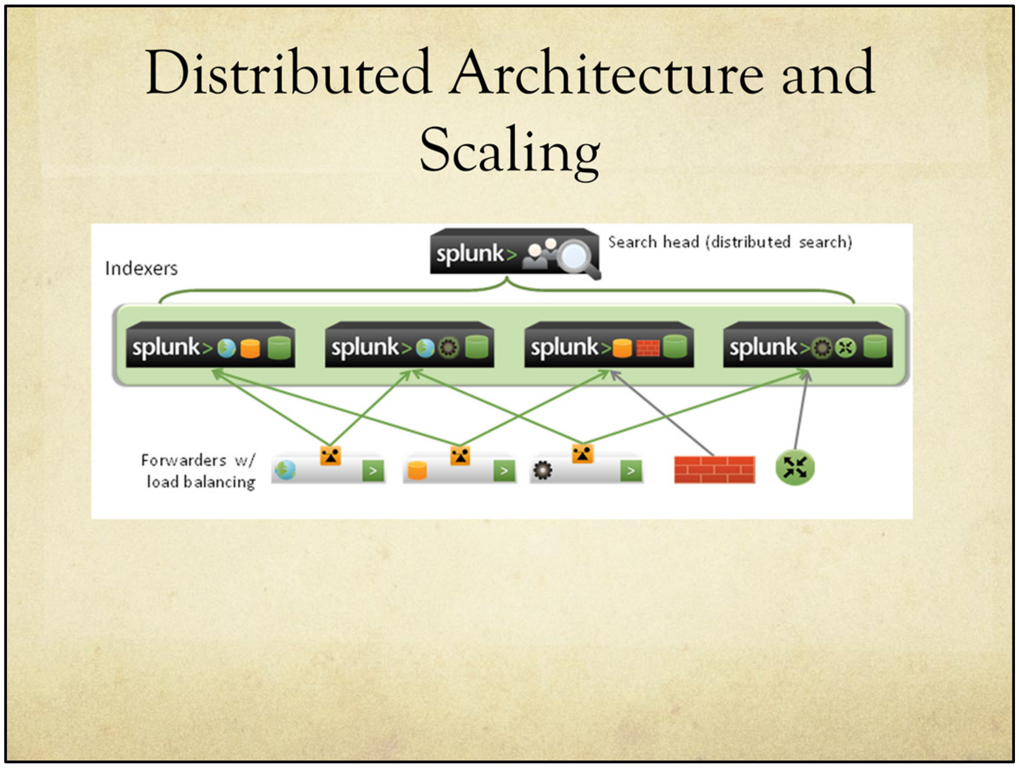


Create Alert

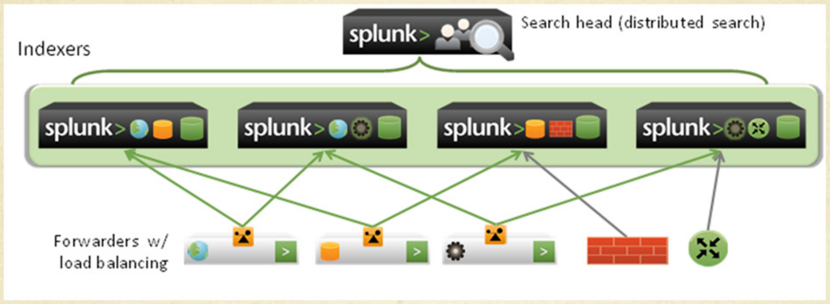
1 Schedule — 2 Actions — 3 Sharing

* Name	
Schedule	Run on a schedule once every...
	Cron schedule
	0 0 * * *
	00 18 * * * (every day at 6 PM) Learn more
	Search time range: -1d to now
	-1d (one day ago), now (triggering time)
	Time specifiers: y, mon, d, h, m, s Learn more
Trigger if	Number of results
	Is less than 500

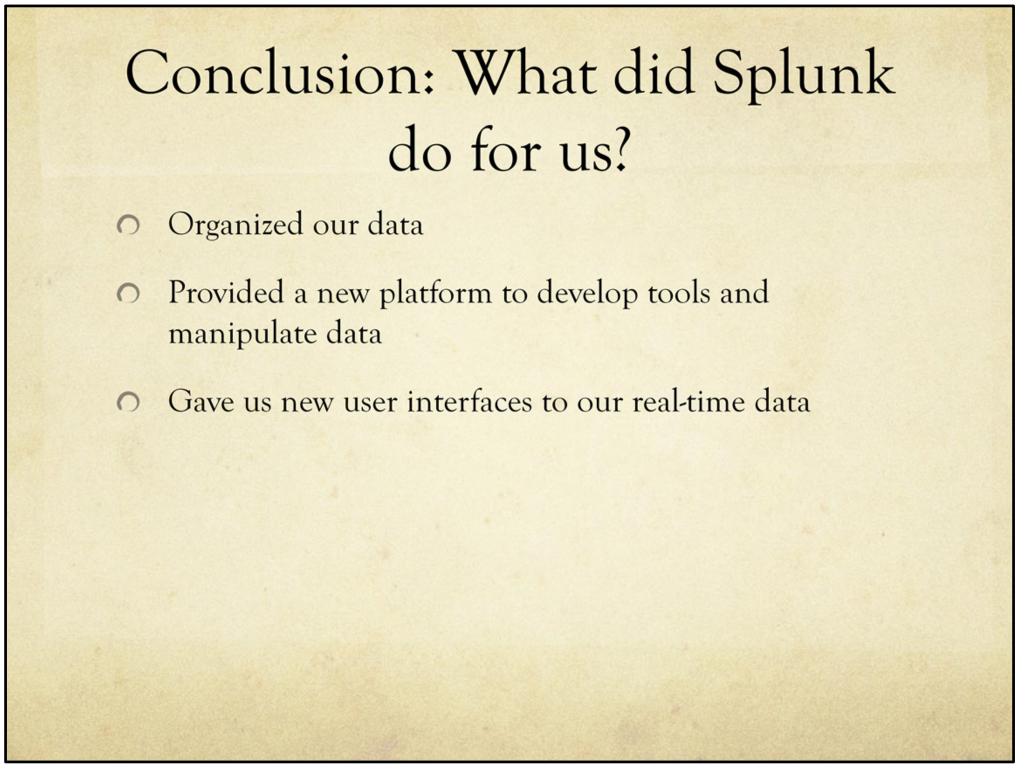
Because searches can be run continuously, Splunk makes it easy to set up an alerting system in case your data isn't showing what you expect it to. One simple example is sending an alert if the spacecraft isn't talking to us when we expect it to. To set this up, you would activate the alert based on some tracking schedule. Then, you specify the search to begin at the beginning of the track, and to trigger if no results have appeared



Distributed Architecture and Scaling


Indexers
Search head (distributed search)
splunk>
splunk>
splunk>
splunk>
splunk>
Forwarders w/ load balancing

Now, you might be wondering what Splunk actually looks like. I won't get too much into detail, but to put it into real terms, Splunk has an architecture that allows it to be distributed across as many machines as you need. Obviously, if you're dealing with terabytes of data a day, one machine won't cut it. By designating many machines to only index data or to only process search queries, you can keep your data behind firewalls or on specific machines without worrying about data privacy issues while simultaneously increasing the throughput of your system.



Conclusion: What did Splunk do for us?

- Organized our data
- Provided a new platform to develop tools and manipulate data
- Gave us new user interfaces to our real-time data